一种基于元胞自动机的动态回溯搜索优化算法 *

杨　军，张达敏†，潘志远，刘　冬，陈娟敏

(贵州大学 大数据与信息工程学院，贵阳 550025)

摘　要：针对传统回溯搜索优化算法存在收敛速度慢、搜索精度不高等问题，提出了一种基于元胞自动机和正交实验设计的改进算法。首先将正交实验设计方法引入算法的交叉算子中，得出具有代表性的优质子代个体；然后在元胞自动机邻居模型的基础上，对个体展开领域内多父代正交交叉操作，提高算法的开采能力和搜索效率；最后对参与交叉的种群引入动态优秀个体比例权重进行选择更新，并采用新的动态变异方程，平衡算法的全局搜索和局部搜索能力。通过对 12 个标准测试函数进行仿真实验，并与其他 6 种表现良好的算法进行比较，结果表明，改进的算法在收敛速度以及寻优精度方面都具有明显优势。

关键词：回溯搜索优化算法；元胞自动机；正交实验设计；多父代正交交叉；动态变异方程

中图分类号：TP301.6　　doi: 10.19734/j.issn.1001-3695.2018.07.0520

Dynamic backtracking search optimization algorithm based on cellular automata

Yang Jun, Zhang Damin†, Pan Zhiyuan, Liu Dong, Chen Juanmin

(*College of Big Data & Information Engineering Guizhou University*, *Guiyang* 550025, *China*)

Abstract: According to the slow convergence speed and low searching precision of traditional backtracking search optimization algorithm, this paper proposed an improved algorithm based on cellular automaton and orthogonal experimental design. Firstly, the algorithm introduced orthogonal experimental design method into the crossover operator to obtain representative high-quality offspring individuals; Then based on the neighbor model of cellular automaton, the method carried out the orthogonal crossover operation of multiple parents in the domain for individuals, which was beneficial to improve the mining capacity and search efficiency of the algorithm; Finally, in order to balance the global searching and local searching ability of the algorithm, the method introduced the dynamic proportional weight of excellent individuals into the cross-population to select and update them, with adopted a new dynamic variation equation. The simulation experiments selected 12 standard test functions and compared with 6 other well-behaved algorithms, the results show that the improved algorithm has obvious advantages in convergence speed and optimization accuracy.

Key words: backtracking search optimization algorithm; cellular automaton; orthogonal experimental design; multiple parent orthogonal crossover; dynamic variation equation

0　引言

智能优化算法是受人类智能、生物群体社会性或自然现象规律的启发而提出的随机搜索算法。因其自身的高效性、通用性和易操作性等特点，近年来在生产调度、图像处理、数据挖掘等众多工程领域得到了广泛的应用。目前常用的智能优化算法主要有遗传算法[1]（genetic algorithm，GA）、粒子群算法[2]（particle swarm optimization，PSO）、差分进化算法[3]（differential evolutionary，DE）、人工蜂群算法[4]（artificial bee colony，ABC）等，以及一些相应的改进算法。例如，Liang 等人[5]提出的粒子群改进算法 CLPSO，夏学文等人[6]改进的 RLPSO；Wang 等人[7]提出的 OXDE，周晓根等人[8]提出的 DELLU；以及 Gao 等人[9]基于蜂群算法改进的 MABC，周新宇等人[10]改进的 ABC-OED 等。

回溯搜索优化算法[11]（backtracking search algorithm，BSA）是 Civicioglu 在 2013 年提出的一种新型启发式优化算法。该算法由于结构框架简单，控制参数较少和独特的记忆功能用于引导搜索，因而具有较强的全局搜索能力，在求解复杂的约束优化问题中发挥了重要作用。

但是，由于 BSA 算法变异算子中的历史种群引导学习，以及交叉算子中两种交叉策略随机调用都具有一定的盲目性。使得算法拥有较好的探索能力，但开发能力较为欠缺，在求解多峰值问题时算法的局部搜索能力较弱并且容易早熟收敛[12-14]。针对这一问题，本文提出一种基于元胞自动机的动态回溯搜索优化算法（dynamic backtracking search optimization algorithm based on cellular automata，CA-DBSA），分别从交叉算子和变异算子入手，在交叉算子中引入了正交实验设计（orthogonal experimental design，OED）方法，并在元胞自动机（cellular automaton，CA）的邻居模型基础上进行并行式多父代正交交叉，较大程度上提高了算法的搜索效率。同时，为了平衡算法的探索与开发能力，在参与交叉的个体选取与变异方程的引导学习环节中分别引入动态自适应调节策略。最后通过对标准测试函数的测试与比较，实验结果表明，改进的算法具有较快的收敛速度

收稿日期：2018-07-12；**修回日期：**2018-08-29　**基金项目：**贵州省自然科学基金项目（黔科合基础［2017］1047 号）

作者简介：杨军（1994-），男，安徽桐城人，硕士研究生，主要研究方向为网络通信、优化计算；张达敏（1967-），男（通信作者），贵州贵阳人，教授，博士，主要研究方向为网络通信、网络拥塞控制（1203813362@qq.com）；潘志远（1994-），女，贵州贵阳人，硕士研究生，主要研究方向为网络通信、优化计算；刘冬（1990-），男，贵州遵义人，硕士研究生，主要研究方向为认知无线网络、优化计算；陈娟敏（1993-），女，贵州晴隆人，硕士研究生，主要研究方向为认知无线网络、优化计算.

与以及较高的寻优精度。

1　回溯搜索优化算法

回溯搜索优化算法是一种基于种群迭代的智能进化算法，与一般的进化算法相似，都具有三个基本的基因算子，即交叉、变异和选择。具体包括五个步骤，分别为种群初始化、选择Ⅰ、种群变异、种群交叉、选择Ⅱ。

1）种群初始化

BSA 的种群初始化以随机产生的方式进行，且包括种群 Pop 和历史种群 $oldPop$ 的同时初始化。即：

$$\begin{cases} Pop_{i,j} \sim U(low_j, up_j) \\ oldPop_{i,j} \sim U(low_j, up_j) \end{cases} \tag{1}$$

其中，$i \in \{1,2,\ldots,SN\}$，$j \in \{1,2,\ldots,D\}$；SN 和 D 分别为种群的规模和维数；low 和 up 分别指搜索空间的下界和上界；U 为随机均匀分布函数。

2）选择Ⅰ

BSA 的选择Ⅰ策略主要用于确定历史种群 $oldPop$ 的更新。如式（2）和（3）所示。

$$oldPop = \begin{cases} Pop & if\ a < b \\ oldPop & otherwise \end{cases} \tag{2}$$

$$oldPop = randperm(oldPop) \tag{3}$$

其中：a 和 b 均为服从区间（0,1）上的均匀分布随机数；$randperm$ 为重新排序函数。在每次迭代过程中，历史种群的个体都有机会取到当代以及之前任意一代的值，凭借其独特的记忆功能引导种群的后续进化。

3）种群变异

BSA 的变异过程如式（4）所示。

$$Mutant = Pop + F \bullet (oldPop - Pop) \tag{4}$$

其中：$Mutant$ 为生成的变异种群；F 为变异尺度系数，用于控制引导搜索向量 $(oldPop - Pop)$ 的幅度，$F = 3 \bullet randn$，$randn$ 为服从标准正态分布的随机数。

4）种群交叉

BSA 的交叉过程包含两种交叉方式，每个个体通过对两种方式的随机调用，生成 0-1 映射矩阵 map，初始 map 为一全 1 矩阵。具体生成公式如式（5）（6）所示：

$$map_{i,u(1:\lceil mixrate \bullet rand \bullet D \rceil)} = 0 \tag{5}$$

$$map_{i,randi(D)} = 0 \tag{6}$$

其中：$u = randperm(D)$，$\lceil\ \rceil$ 表示向上取整，$mixrate$ 为设置的交叉概率，$rand$ 为生成一个 $[0,1]$ 区间的随机数；$randi(D)$ 为生成一个 $[1, D]$ 区间的随机整数。

再由确定的 map 映射矩阵，实现种群 Pop 和变异种群 $Mutant$ 之间的对应交叉，最终产生新的实验种群 T。如式（7）所示。

$$T_{i,j} = \begin{cases} Pop_{i,j} & map_{i,j} = 0 \\ Mutant_{i,j} & map_{i,j} = 1 \end{cases} \tag{7}$$

5）选择Ⅱ

BSA 的选择Ⅱ策略采用了贪婪选择机制，在新生成的实验种群 T 和种群 Pop 之间选择对应位置上的最优（适应度值较小的）个体用于下一轮进化。如式（8）所示。

$$Pop_i = \begin{cases} T_i & if\ \ fitness(T_i) < fitness(Pop_i) \\ Pop_i & otherwise \end{cases} \tag{8}$$

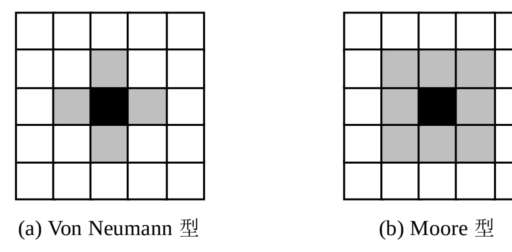
重复以上步骤，直至满足循环终止条件，最后输出最优解。

2　元胞自动机与正交实验设计

2.1　元胞自动机

元胞自动机[15~17]（CA）是由 Neumann 首先提出的一种经典的自然计算模型，具有模拟复杂系统时空演化过程的能力。CA 共由元胞、状态、邻居和规则四部分组成，是一种时间、空间和状态都具有离散性的动力学系统。所有的元胞构成一个元胞空间，元胞空间中的每个元胞将并行同步演变；元胞在某一时刻只能存在一种状态，且该状态来源于固定的有限离散状态集；邻居是元胞个体周围按一定几何规则划分的元胞集合，每个元胞的下一时刻状态由该元胞以及与其对应的邻居共同决定；规则是根据元胞当前状态及其邻居状态确定下一时刻该元胞状态的动力学函数。

二维元胞自动机的邻居模型划分通常可分为四种：冯-诺依曼型，摩尔型，扩展的摩尔型，马哥勒斯型。其中最常用的是冯-诺依曼（Von Neumann）型和摩尔（Moore）型，以常用规则四方网格划分为例，如图 1 所示。



(a) Von Neumann 型　　(b) Moore 型

图 1　常用元胞自动机邻居模型

Fig. 1　Common cellular automaton neighbor models

由于元胞空间理论上在各维上无限延伸。因此，对于二维元胞空间需定义周期型边界条件，即上下相接，左右相接，形成一个拓扑的圆环面。如图 2 所示。

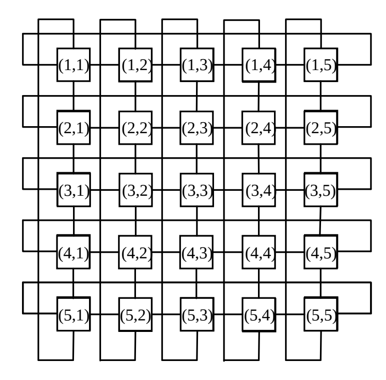


图 2　二维元胞空间拓扑结构

Fig. 2　2-dimensional cellular spatial topology

2.2　正交实验设计

正交实验设计[17~19]（OED）是一种利用正交表来解决多因素、多水平实验的有效方法。在给定的实验范围内，通过正交性准则选取合适的正交表 $L_M(Q^N)$，来安排较少的实验找出最优的水平组合方式。$L_M(Q^N)$ 表示 N 因素，Q 水平的正交表，M 表示需要安排的实验次数。当 N 和 Q 较大时，执行全面实验的次数为 Q^N，而 OED 方法只需从 Q^N 次全面实验中挑选出 M 个均匀分散且具有代表性的点进行实验（M 通常远小于 Q^N），较大程度的提高了工作效率。其中，$M = Q^J$，Q 为素数，J 为满足式（9）的正整数。

$$N \le \frac{Q^J - 1}{Q - 1} \tag{9}$$

例如一个四因素，三水平的实验，可以通过正交表 $L_9(3^4)$ 来执行，如式（10）。但若进行全面实验则需 81（34）次。

$$L_9(3^4)=\begin{bmatrix}1&1&1&1\\1&2&2&2\\1&3&3&3\\2&1&2&3\\2&2&3&1\\2&3&1&2\\3&1&3&2\\3&2&1&3\\3&3&2&1\end{bmatrix} \tag{10}$$

3　改进的 BSA

3.1　基于元胞自动机的多父代正交交叉算子

基本回溯搜索算法采用的交叉方式是常用的二项式交叉，而二项式交叉算子随机产生子代个体，并不能实现对相应搜索空间的充分开采。多父代正交交叉算子[18~20]利用正交实验设计的方法，对多个父代进行因素和水平的划分，并采用正交表产生一系列具有代表性的组合来生成子代个体，从而在确定的样本空间中完成高效搜索。本文结合元胞自动机模型，提出一种基于元胞自动机的多父代正交交叉算子。该算子建立在 Von Neumann 型元胞邻居基础上，每个个体与周围的四邻居共同组成多父代参与正交交叉，最后选出子代中最优的水平组合个体作为该元胞的下一时刻状态。由于元胞自动机的并行演变策略和正交交叉的高效搜索性能，使得该新型交叉算子在寻优速度和精度上都具有较大的提升。

同时，为了避免算法后期陷入局部最优而早熟收敛的可能性，在交叉算子中引入了随机扰动策略。当元胞个体满足交叉概率时执行多父代正交交叉；反之，则对该个体随机挑选任意维以初始化的方式进行扰动。进一步增强了种群的多样性。

算子操作具体描述如下：

a）判断元胞个体是否满足交叉概率 $mixrate$ 。如果 $rand<mixrate$ 成立则跳转至 b)，执行多父代正交交叉。否则对该个体进行随机扰动：

$$dim=ceil(rand\bullet D) \tag{11}$$

$$x_{i,j}=\begin{cases}low_j+(up_j-low_j)\bullet rand & if\ \ j\in randperm(D,dim)\\ x_{i,j} & otherwise\end{cases} \tag{12}$$

b)将参与交叉的 Q 个父代个体分别作为正交实验设计的一个水平，记第 i 个水平为 β_i ， $i\in\{1,2,\ldots,Q\}$ ，则 $\beta_i=x_i$ ， $x_i\in\{x_1,x_2,\ldots,x_Q\}$ 。

c)当优化问题的维度较小时，可直接将因素与维度进行匹配；但是通常在复杂的高维约束优化问题中，因素的选取则需对维度进行切片划分，从而降低 OED 的工作量。假定将参与交叉的任一个体 x_i 分成 $E(E=N)$ 个片段，每个片段即对应一个因素，如式（13）所示。

$$\begin{cases}G_1=(x_1,\ldots,x_{k_1})\\G_2=(x_{k_1+1},\ldots,x_{k_2})\\\ldots\\G_E=(x_{k_{E-1}+1},\ldots,x_D)\end{cases} \tag{13}$$

其中：$k_1,k_2,\ldots,k_{E-1}$ 为 $(1,D)$ 之间随机生成的正整数，且满足 $1<k_1<k_2<\ldots<k_{E-1}<D$ 。

d)令 $k_0=0$, $k_E=N$,因此第 j $(j=1,2,\ldots,N)$ 个因素的 Q 个水平可以表示为

$$\begin{cases}G_j(1)=(\beta_{1,k_{j-1}+1},\beta_{1,k_{j-1}+2},\ldots,\beta_{1,k_j})\\G_j(2)=(\beta_{2,k_{j-1}+1},\beta_{2,k_{j-1}+2},\ldots,\beta_{2,k_j})\\\ldots\\G_j(Q)=(\beta_{Q,k_{j-1}+1},\beta_{Q,k_{j-1}+2},\ldots,\beta_{Q,k_j})\end{cases} \tag{14}$$

e)构造正交表 $L_M(Q^N)=[a_{i,j}]_{M\times N}$ ，根据正交表 $L_M(Q^N)$ 安排实验，产生 M 个子代个体（ M 组实验）：

$$\begin{cases}(G_1(a_{1,1}),G_2(a_{1,2}),\ldots,G_N(a_{1,N}))\\(G_1(a_{2,1}),G_2(a_{2,2}),\ldots,G_N(a_{2,N}))\\\ldots\\(G_1(a_{M,1}),G_2(a_{M,2}),\ldots,G_N(a_{M,N}))\end{cases} \tag{15}$$

f）采用式（16）计算分析每种因素的每个水平对实验结果的影响，从而得出最优的水平组合方式。

$$S_{nq}=\frac{\sum_{m=1}^{M}F_m\times z_{mnq}}{\sum_{m=1}^{M}z_{mnq}} \tag{16}$$

其中：F_m 为第 $m(m=1,2,\ldots,M)$ 组实验的结果， S_{nq} 为因素 $n(n=1,2,\ldots,N)$ 的第 $q(q=1,2,\ldots,Q)$ 个水平对结果的影响程度。 z_{mnq} 表示第 m 组实验中是否包含因素 n 的第 q 个水平，若包含则 $z_{mnq}=1$ ， 否则 $z_{mnq}=0$ 。

3.2　交叉选择机制

BSA 算法的交叉环节中，通过 map 映射矩阵完成对变异种群 $Mutant$ 和种群 Pop 的随机交叉操作。虽然已将原始交叉替换为基于元胞自动机的多父代正交交叉算子，较好地提高了算法的搜索效率和精度，但在搜索方向上仍然存在一定的盲目性。为此，受遗传算法交叉前进行一定概率的择优选取启发[1]，本文引入交叉选择机制，并提出一种动态优秀个体比例权重 $p\%$ 。用于对参与交叉的个体进行选择，在算法执行的不同阶段以合适的搜索方向引导交叉。

当种群变异结束后，变异种群 $Mutant$ 和种群 Pop 将作为候选交叉成员。从候选成员中选取适应度值最优的前 $SN\bullet p\%$ 个个体，再从剩下的成员中随机选取 $SN\bullet(1-p\%)$ 个个体共同组成交叉群体。优秀个体比例权重 $p\%$ 满足如下公式：

$$p=p_{\min}+(p_{\max}-p_{\min})\bullet\left(\frac{epk}{epoch}\right)^2 \tag{17}$$

其中: $p_{\min}$ 和 $p_{\max}$ 分别为权重的最小值和最大值； epk 和 $epoch$ 分别为算法的当前迭代次数和迭代总次数。由式（17）可以看出，随着进化代数的不断增加，优秀个体比例权重 $p\%$ 越来越大。使得算法在执行前期交叉算子偏向于全局搜索，而到了中后期时则偏向于在全局最优值附近进行精确的局部搜索。在搜索方向上起到了较好的引导作用。

3.3　动态变异方程

通过对 BSA 算法的变异算子分析可知，该算法的变异操作能记忆之前迭代过的种群，并以迭代过的历史种群作为经验来指导当前种群的搜索方向，使之具有较强的全局搜索能力。但因其仅仅通过设置历史种群对搜索方向进行引导，个体进化缺乏一定的学习能力，导致算法的收敛速度减慢，局部开采能力也较弱。针对这一缺点，在文献[5,13]提出的广泛学习策略以及 ABC 算法雇佣蜂搜索方程[4,9]的基础上，本文在 BSA 的变异过程中同时加入最优个体和较优群体的引导，提出一种带有加权因子的自适应动态变异方程。公式如式（18）（19）所示。

$$\begin{aligned}Mutant_i=&u\left(Pop_i+F\bullet(oldPop_{r_1}-Pop_{r_2})\right)+\\&(1-u)\left(Pop_{best}+F\bullet(oldPop_{gr_best}-Pop_i)\right)\end{aligned} \tag{18}$$

$$u = \exp\left(-\sqrt{epk} - 1\right) \tag{19}$$

其中：u 为加权因子，i，r_1，r_2 为 $[1, SN]$ 上互不相等的随机整数；Pop_{best} 为当前种群搜索到的全局最优个体，$oldPop_{gr_best}$ 为当代历史种群中前 $t(\mathrm{mod}(SN,t)=0)$ 个优秀个体组成的较优群体。选出的种群 $oldPop_{gr_best}$ 和种群 $oldPop$ 规模相同，具体操作如下：将 $oldPop$ 中选出的前 t 个优秀个体复制成 SN/t 组，并对每一组随机排序，最后重新组合成种群 $oldPop_{gr_best}$ 。

式（18）的前半部分通过选取种群 $oldPop$ 和 Pop 中与变异个体 i 不同的个体组成差分向量来增强算子的勘探能力，后半部分通过全局最优个体 Pop_{best} 和较优历史群体 $oldPop_{gr_best}$ 来增强算子的精英学习能力和开采能力。并且利用式（19）[21]在算法执行的不同阶段对全局搜索和局部搜索能力进行合适的加权。使得算法执行前期以全局搜索为主，而后期则以局部开发为主。

3.4　改进算法流程

改进后的 CA-DBSA 算法具体实现流程如下：

a)初始化种群 Pop 、历史种群 $oldPop$ 以及种群规模 SN 、交叉概率 $mixrate$ 和最大进化代数 $epoch$ 等参数，设置当前进化代数 $epk=1$ 。

b)执行选择Ⅰ策略：更新历史种群 $oldPop$ 。

c)评价种群 Pop 和历史种群 $oldPop$ 的适应度值，选出最优个体 Pop_{best} 和较优历史群体 $oldPop_{gr_best}$ 。

d)执行变异操作：根据式（18）（19）计算出加权因子 u 和变异种群 $Mutant$ 。

e)执行交叉选择机制：根据（17）计算出当代优秀个体比例权重 $p\%$，并从种群 Pop 和变异种群 $Mutant$ 中完成对参与交叉个体的选择。

f)将选出的参与交叉群体随机分布在规模为 $a \times a\,(a \times a = SN)$ 的二维网格元胞空间中。

g)执行交叉操作：以交叉概率 $mixrate$ 对元胞空间中的每个元胞个体进行判断。如果满足则以 Von Neumann 型元胞邻居结构为基础，构造正交表 $L_M\left(Q^N\right)$ 执行多父代正交交叉，并获得最优组合形式的子代。否则以式（11）（12）对个体进行随机扰动。最终得出实验种群 T 。

h)执行选择Ⅱ策略：评价实验种群 T ，选出用于下一轮进化的优秀种群 Pop 。

i)找出第 epk 代的最优个体和最优适应度值。

j)判断是否达到循环终止条件。若是则转步骤 k)，否则 $epk = epk + 1$，跳转至步骤 b)。

k)输出当前的最优个体和最优适应度值。

4　数值实验及结果分析

为了验证本文算法的可行性与效果，选取 12 个标准测试函数[68]对其进行数值实验。同时与基本 BSA 算法[11]、参考文献中的 CLPSO[5]、MABC[9]和 LBSA[14]算法，以及利用正交交叉策略改进的 OXDE[7] 和多父代交叉算子的 MPCO-ABC[20]算法进行性能比较。

测试函数如表 1 所示。表中给出了函数具体名称和表达式、搜索空间范围以及最优值。其中 $f_1 \sim f_5$ 是单峰类型函数，侧重于衡量算法的收敛速度和精度；$f_6 \sim f_{12}$ 是多峰类型函数，侧重于衡量算法的全局搜索能力。

表 1　12 个标准测试函数

Table 1　12 standard test functions

函数名	函数表达式	搜索空间	最优值
Sphere	$f_1(x)=\sum_{i=1}^{n} x_i^2$	[-100,100]	0
Tablet	$f_2(x)=10^6 x_1^2+\sum_{i=2}^{n} x_i^2$	[-100,100]	0
Rosenbrock	$f_3(x)=\sum_{i=1}^{n-1}\left(100\left(x_{i+1}-x_i^2\right)^2+(1-x_i)^2\right)$	[-30,30]	0
Schwefel 1.2	$f_4(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j^2\right)^2$	[-100,100]	0
Schwefel 2.22	$f_5(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	[-10,10]	0
Ackley	$f_6(x)=20+e-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)$	[-32,32]	0
Alpine	$f_7(x)=\sum_{i=1}^{n}\lvert x_i\sin(x_i)+0.1x_i\rvert$	[-10,10]	0
Rastrigin	$f_8(x)=\sum_{i=1}^{n}\left(x_i^2-10\cos(2\pi x_i)+10\right)$	[-5.12,5.12]	0
Griewank	$f_9(x)=\sum_{i=1}^{n}\frac{x_i^2}{4000}-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600,600]	0
Schwefel 2.26	$f_{10}(x)=418.98288727216249D-\sum_{i=1}^{n}\left(x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	[-500,500]	0
Levy and Montalvo 1	$f_{11}(x)=\frac{\pi}{n}\left(10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\left(1+10\sin^2(\pi y_i+1)\right)+(y_n-1)^2\right), y_i=1+0.25(x_i+1)$	[-10,10]	0
Levy and Montalvo 2	$f_{12}(x)=0.1\left(\sin^2(3\pi x_i)+\sum_{i=1}^{n-1}(x_i-1)^2\left(1+\sin^2(3\pi x_{i+1})\right)+(x_n-1)^2\left(1+\sin^2(2\pi x_n)\right)\right)$	[-5,5]	0

本次实验中，7 种对比算法的种群规模 *SN* 和最大进化代数 *epoch* 均设置为 25 和 5000；基本 BSA 算法的交叉概率 *mixrate* 设为 1；CA-DBSA 中优秀个体比例权重 p_{min} 、 p_{max} 分别为 10、90；正交表采用六因素五水平的 $L_{25}(5^6)$ 。其他算法的参数设置方法按照文献[5,7,9,14,20]进行对应设置。

实验分为三组进行，第 1、2 组实验中，测试函数的维数 *D* 均取 30；第 3 组实验中 *D* 均取 60。针对同一测试函数的仿真实验，每种算法独立运行 30 次，记录其寻得最优值的平均值（Mean）和标准差（Std.Dev）。为了进一步分析比较 7 种算法之间的性能差异，在第 2、3 组实验中引入显著性水平 $\alpha=0.05$ 的 Wilcoxon 秩和检验方法对其结果进行直观呈现。+、-、≈分别表示其他算法的性能要优于、劣于、相当于 CA-DBSA 算法。

为了证明本文所提算法 CA-DBSA 交叉概率 *mixrate* 的最理想取值，在第一组实验中，对 *mixrate* 分别取 0.5~0.9 的 5 个不同的值进行测试。结果如表 2 所示，其中每个函数的最优测试结果已对其进行加粗体现。从最优实验结果的分布情况来看，当交叉概率 *mixrate* 取值为 0.7 时，CA-DBSA 算法的寻优能力在 12 个测试函数上均表现最好。当交叉概率偏小时，算法开采能力较弱，局部搜索不充分。反之，在偏大时则易陷入局部最优。因此，在接下来的第二和第三组实验中，本文的 CA-DBSA 算法交叉概率 *mixrate* 均取值为 0.7。

表 3 给出了第 2 组实验 $D=30$ 时各算法的测试结果。从表中每个函数的最优测试结果可以看出，共有 9 个函数的最优结果是由 CA-DBSA 算法实验得到。表明了本文的 CA-DBSA 算法在大部分测试函数上的实验结果都要优于其他几种算法，表现出了较强的全局搜索和局部搜索能力。相对于改进的 CLPSO、MABC 和基本 BSA 来说，CA-DBSA 算法在寻优精度和稳定性上均具有明显的优势；MPCO-ABC 和 LBSA 均分别只在一个函数上性能优于 CA-DBSA；而对于 OXDE，只有在函数 f_3 、 f_6 和 f_{11} 上，OXDE 算法的寻优能力比本文算法更优，且 f_9 上相当于本文算法。这正也体现出了正交实验设计方法的有效性。

表 2 时 ca-dbsa 算法 *mixrate* 不同取值的实验结果

Table 2 CA-DBSA algorithm with D=30 experimental results with different values of mixrate

函 数	Mean±Std.Dev				
	mixrate =0.5	*mixrate* =0.6	*mixrate* =0.7	*mixrate* =0.8	*mixrate* =0.9
f_1	2.82E-114±3.41E-114	7.03E-118±3.75E-117	**3.56E-121±1.38E-120**	2.29E-120±3.42E-119	1.63E-116±5.97E-117
f_2	1.17E-112±5.26E-110	1.85E-117±2.44E-116	**1.60E-122±4.73E-122**	3.04E-119±3.26E-119	6.05E-115±3.09E-116
f_3	7.44E+01±5.52E+01	5.81E+01±2.47E+01	**1.81E+00±1.56E-01**	2.23E+00±1.07E+00	3.96E+01±1.20E+01
f_4	5.41E-158±3.11E-160	2.63E-165±2.79E-164	**1.47E-174±3.10E-174**	1.83E-171±4.36E-171	1.47E-164±4.72E-166
f_5	3.02E-63±1.48E-63	4.92E-68±1.89E-66	**1.37E-75±8.77E-75**	5.26E-72±7.25E-71	2.32E-66±6.74E-66
f_6	9.63E-09±9.91E-08	5.03E-11±4.99E-10	**2.03E-12±4.50E-12**	4.15E-10±3.89E-10	3.37E-09±4.68E-10
f_7	4.81E-11±6.23E-12	3.47E-12±1.16E-12	**7.94E-15±1.03E-15**	5.11E-14±1.57E-13	2.87E-12±4.15E-11
f_8	6.05E-01±4.67E+00	4.76E-01±2.67E-01	**2.05E-01±3.42E-02**	8.49E-01±6.64E-01	5.29E+00±1.10E+00
f_9	1.93E-07±2.56E-06	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	2.05E-10±1.35E-11	2.29E-08±4.51E-08
f_{10}	5.83E-09±2.26E-10	1.52E-11±1.47E-11	**1.82E-12±1.13E-13**	9.01E-10±7.23E-11	7.05E-10±4.43E-10
f_{11}	2.76E-16±3.71E-16	4.73E-19±1.05E-20	**1.49E-21±1.75E-22**	1.19E-18±1.32E-18	3.54E-17±2.16E-16
f_{12}	6.87E-24±3.61E-26	3.13E-27±3.65E-28	**1.13E-31±2.68E-31**	3.24E-29±4.41E-29	1.17E-25±2.09E-26

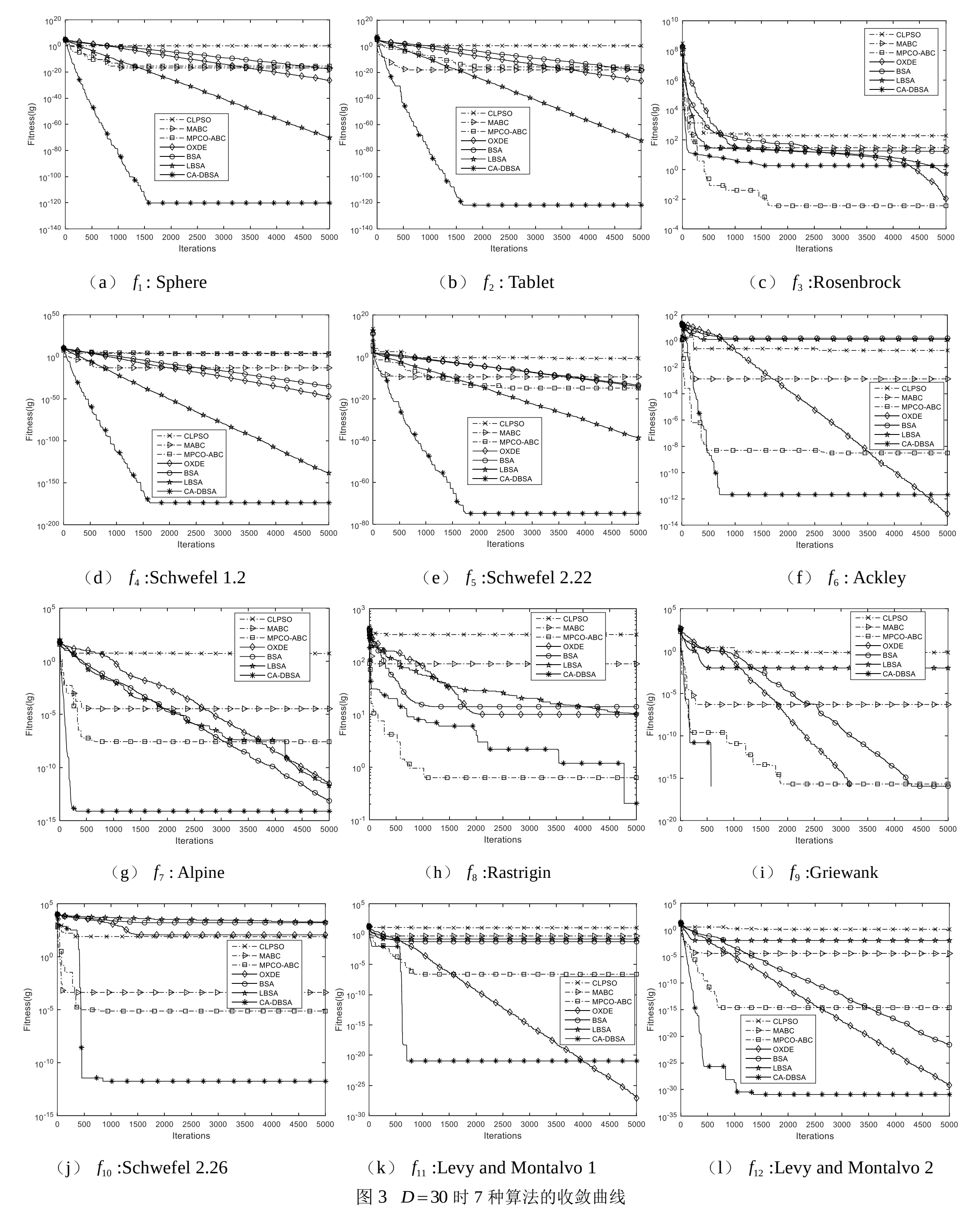
表 3 时 7 种算法的实验结果

Table 3 Experimental results of 7 algorithms at D=30

函 数		CLPSO	MABC	MPCO-ABC	OXDE	BSA	LBSA	CA-DBSA
f_1	Mean	2.43E+00	1.42E-17	3.69E-16	2.02E-26	9.36E-20	5.88E-71	**3.56E-121**
	Std.Dev	1.92E-01	6.89E-17	5.68E-15	1.23E-25	4.64E-20	3.32E-68	**1.38E-120**
f_2	Mean	5.90E+00	1.13E-18	1.58E-16	2.79E-27	2.51E-18	4.45E-73	**1.60E-122**
	Std.Dev	8.46E+00	4.84E-18	2.91E-14	9.11E-26	2.24E-18	2.24E-72	**4.73E-122**
f_3	Mean	1.24E+02	2.88E+01	**3.60E-03**	1.11E-02	2.50E+01	1.30E-01	1.81E+00
	Std.Dev	5.23E+02	1.27E+00	**8.03E-03**	1.34E-02	3.21E+01	2.52E-01	1.56E-01
f_4	Mean	1.09E+04	6.89E-14	5.09E+03	1.58E-48	4.00E-36	4.13E-139	**1.47E-174**
	Std.Dev	6.04E+03	2.37E-15	3.81E+03	4.65E-48	4.73E-35	5.68E-137	**3.10E-174**
f_5	Mean	1.79E-01	2.40E-10	1.20E-15	7.75E-15	3.14E-14	2.12E-39	**1.37E-75**
	Std.Dev	3.47E-01	6.22E-11	7.34E-15	1.26E-16	1.52E-12	1.74E-39	**8.77E-75**
f_6	Mean	1.95E-01	1.35E-03	3.06E-09	**7.19E-14**	1.65E+00	1.34E+00	2.03E-12
	Std.Dev	1.61E-01	1.33E-03	1.26E-10	**5.77E-15**	1.34E+00	2.86E-01	4.50E-12
f_7	Mean	5.38E+00	3.80E-05	2.69E-08	3.24E-12	7.44E-14	1.92E-12	**7.94E-15**
	Std.Dev	2.18E-01	2.54E-07	2.12E-08	4.65E-13	1.42E-14	4.78E-12	**1.03E-15**
f_8	Mean	3.21E+02	9.02E+01	6.29E-01	9.95E+00	1.39E+01	1.03E+01	**2.05E-01**
	Std.Dev	9.16E+01	1.50E+00	1.07E+01	6.03E-01	2.19E+01	1.48E+00	**3.42E-02**
f_9	Mean	6.75E-01	4.92E-07	2.01E-16	**0.00E+00**	1.11E-16	9.90E-03	**0.00E+00**
	Std.Dev	1.28E-02	9.16E-08	4.43E-16	**0.00E+00**	4.47E-16	6.45E-01	**0.00E+00**
f_{10}	Mean	8.02E+01	4.32E-04	7.59E-06	1.18E+02	1.66E+03	1.95E+03	**1.82E-12**

	Std.Dev	1.52E+01	4.09E-03	6.75E-06	7.11E+02	1.05E+03	2.29E+01	**1.13E-13**
f_{11}	Mean	1.04E+01	4.95E-01	2.36E-07	**8.15E-28**	5.70E-02	1.52E-01	1.49E-21
	Std.Dev	1.12E+01	1.60E+00	2.28E-06	**3.83E-28**	3.77E-03	1.27E-01	1.75E-22
f_{12}	Mean	1.36E+00	3.60E-05	2.55E-15	6.55E-30	2.65E-22	1.10E-02	**1.13E-31**
	Std.Dev	1.38E+00	2.09E-04	1.94E-16	1.02E-29	1.42E-21	6.24E-02	**2.68E-31**
	+	0	0	1	3	0	1	
	-	12	12	11	8	12	11	
	≈	0	0	0	1	0	0	

除了实验结果的寻优精度和稳定性，收敛速度也是衡量算法能力的一项重要指标。图 3 给出了 D=30 时各算法在 12 个测试函数上优化过程的收敛曲线。为了更清晰地描述收敛过程中曲线的变化，图中的纵坐标取适应度值的对数。从图中可以看出，在没有早熟收敛的前提下，与其他 6 种算法相比，CA-DBSA 算法在 12 个函数上整体来说均表现出了较快的收敛速度。除了在函数 f_8 上，本文算法迭代到 5000 代时仍未完全收敛，在其他函数上均在前半周期已经收敛。这得益于基于元胞自动机的多父代正交交叉算子的使用，使得 CA-DBSA 在搜索效率上相比于基本 BSA 得到了较大程度的提高。



（a） f_1: Sphere　（b） f_2: Tablet　（c） f_3 :Rosenbrock

（d） f_4 :Schwefel 1.2　（e） f_5 :Schwefel 2.22　（f） f_6 : Ackley

（g） f_7 : Alpine　（h） f_8 :Rastrigin　（i） f_9 :Griewank

（j） f_{10} :Schwefel 2.26　（k） f_{11} :Levy and Montalvo 1　（l） f_{12} :Levy and Montalvo 2

图 3　D=30 时 7 种算法的收敛曲线

Fig. 3　Convergence curves of 7 algorithms at D=30

对于智能进化算法而言，随着优化问题维度的不断增加，问题的复杂度也将会随之急剧上升。而对于多峰类型函数来说，其局部极值个数将会呈现指数级别的增长。因此算法的整体性能将会受到较大影响。在此基础上，为了进一步研究本文所提算法受维度增长的影响程度，对 7 种算法在 12 个测试函数上进行了第三组 $D=60$ 时的数值实验。实验结果如表 4 所示。从表中可以看出，当测试函数的维数从 30 增加到 60 时，相比于表 3 中的实验结果，各算法的收敛性能均有所下降，但是 CA-DBSA 受到的影响相对其他 6 种算法来说较小。且从表中加粗部分数据结果和秩和检验的统计信息来看，12 个函数的最优测试结果中有 10 个是由 CA-DBSA 算法实验所得，比表 3 实验结果表现更好。在 OXDE 中，只有在函数 f_6 和 f_9 上，OXDE 的寻优结果比 CA-DBSA 更好。而表 3 中存在 4 个结果比 CA-DBSA 更优或相似。同样，30 维时的 LBSA 有一个函数寻优性能优于 CA-DBSA，而在 60 维时性能有所下降。因此，与表 3 对比可发现，本文算法在 60 维度时的整体实验结果比 30 维度时有所提高，展现出了较强的鲁棒性。

表 4 时 7 种算法的实验结果

Table 4 Experimental results of 7 algorithms at D=60

函数		CLPSO	MABC	MPCO-ABC	OXDE	BSA	LBSA	CA-DBSA
f_1	Mean	1.33E+02	7.27E-07	5.51E-09	7.07E-11	6.63E-06	2.07E-24	**1.33E-52**
	Std.Dev	4.17E+01	9.89E-08	4.62E-08	7.90E-11	3.90E-05	1.52E-26	**7.29E-52**
f_2	Mean	1.46E+02	2.37E-06	9.04E-08	5.27E-10	7.01E-06	5.96E-27	**5.77E-52**
	Std.Dev	2.25E+00	5.76E-05	6.61E-08	1.19E-11	9.96E-06	3.62E-28	**1.20E-53**
f_3	Mean	3.13E+04	5.73E+04	7.48E+02	1.56E+02	2.41E+02	4.02E+02	**8.73E+01**
	Std.Dev	1.94E+03	1.41E+04	4.23E+01	2.54E+02	2.23E+02	3.68E+01	**3.48E+00**
f_4	Mean	1.44E+06	4.29E+02	5.16E+05	2.91E-19	2.40E-09	7.62E-31	**5.81E-69**
	Std.Dev	6.59E+03	1.28E+02	6.49E+04	5.64E-20	3.47E-09	1.26E-28	**1.60E-70**
f_5	Mean	5.02E+03	2.72E-04	7.22E-07	4.94E-05	8.79E-04	3.79E-08	**4.28E-35**
	Std.Dev	7.74E+01	2.21E-06	6.36E-08	8.80E-05	6.50E-03	1.17E-06	**1.15E-34**
f_6	Mean	2.04E+01	1.52E+01	2.67E-03	**2.52E-06**	5.58E+00	6.40E+02	2.18E-02
	Std.Dev	2.01E+01	8.05E+00	1.83E-04	**1.50E-05**	3.25E-02	3.31E+02	2.66E-04
f_7	Mean	1.08E+02	6.43E-01	8.04E-05	9.78E-07	1.19E-04	8.14E-09	**1.83E-14**
	Std.Dev	1.05E+02	6.78E-01	1.62E-03	1.52E-08	3.76E-05	6.22E-08	**1.52E-16**
f_8	Mean	7.87E+02	1.75E+02	4.37E+01	5.17E+01	4.48E+01	8.94E+01	**2.07E+01**
	Std.Dev	1.03E+01	7.58E+02	2.03E+00	6.17E+01	4.74E+01	6.25E+00	**2.56E+00**
f_9	Mean	8.72E+02	2.47E-03	2.41E-06	**3.11E-11**	9.87E-03	1.30E+01	5.13E-10
	Std.Dev	3.07E+02	5.95E-04	3.72E-06	**3.36E-10**	1.62E-01	5.34E+01	7.12E-10
f_{10}	Mean	1.63E+04	1.05E+04	7.81E+02	3.77E+03	3.57E+03	7.71E+03	**1.37E+02**
	Std.Dev	1.87E+04	8.87E+03	6.33E+02	3.57E+03	3.95E+03	5.22E+03	**2.37E+00**
f_{11}	Mean	1.66E+02	1.85E+00	5.18E+02	7.10E-02	2.19E-01	1.06E+01	**1.25E-05**
	Std.Dev	1.71E+01	7.51E-01	1.58E+01	9.98E-02	2.23E-01	2.05E+00	**4.69E-04**
f_{12}	Mean	4.14E+01	5.89E-01	1.73E-09	6.33E-14	3.28E-08	1.49E+01	**8.71E-18**
	Std.Dev	3.60E+01	3.23E-02	3.57E-07	1.26E-13	2.10E-07	6.37E+02	**7.11E-18**
	+	0	0	1	2	0	0	
	-	12	12	11	10	12	12	
	≈	0	0	0	0	0	0	

5 结束语

基本 BSA 算法具有较强的全局搜索能力，但是局部搜索能力较为欠缺，且收敛速度较慢、寻优精度不高。本文对 BSA 的变异和交叉操作进行改进，提出一种新的基于元胞自动机的多父代正交交叉算子，有效提高了算法的搜索效率和精度。并在交叉过程中引入了随机扰动策略，避免了算法执行中后期易陷入局部最优的可能。同时，提出一种动态变异方程，并在交叉前期执行交叉选择机制，在算法进化的不同阶段较好的平衡了整体的全局和局部搜索能力。最后，通过选取的标准函数和其他 6 种算法进行数值实验。结果表明，本文改进的算法相对基本 BSA 和参考文献中的 5 种算法而言，在收敛速度和精度上都具有明显的优势，搜索性能得到了较大的提升。

参考文献：

[1] Holland J H. Adaptation in natural and artificial systems [M].Cambridge: MIT Press, 1992.

[2] Kennedy J, Eberhart R. Particle swarm optimization [C]//Proc of IEEE International Conference on Neural Networks.Piscataway,NJ:IEEE Press, 1995: 1942-1948.

[3] Storn R, Price K. Differential evolution:a simple and efficient heuristic for global optimization over continuous spaces [J]. Journal of Global Optimization, 1997, 11(3): 341-359.

[4] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm [J]. Journal of Global Optimization, 2007, 39(3): 459-471.

[5] Liang Jing, Qin Kai, Suganthan P N, *et al.* Comprehensive learning particle swarm optimizer for global optimization of multimodal functions [J]. IEEE Trans on Evolutionary Computation, 2006, 10(3): 281-295.

[6] 夏学文，刘经南，高柯夫,等.具备反向学习和局部学习能力的粒子群算法 [J]. 计算机学报，2015, 38(7): 1397-1407. (Xia Xuewen, Liu Jingnan, Gao Kefu, *et al.* Particle swarm optimization algorithm with

reverse-learning and local-learning behavior [J]. Chinese Journal of Computers, 2015, 38(7): 1397-1407.)

[7] Wang Yong, Zhang Qingfu. Enhancing the search ability of differential evolution through orthogonal crossover [J]. Information Sciences, 2012, 185(1): 153-177.

[8] 周晓根，张贵军，郝小虎,等. 一种基于局部 Lipschitz 下界估计支撑面的差分进化算法 [J]. 计算机学报, 2016,39(12): 2631-2651. (Zhou Xiaogen, Zhang Guijun, Hao Xiaohu, *et al*. Differential evolution algorithm based on local Lipschitz underestimate supporting hyperplanes [J].Chinese Journal of Computers,2016,39(12): 2631-2651.)

[9] Gao Weifeng, Liu Sanyang. A modified artificial bee colony algorithm [J]. Computers & Operations Research, 2012, 39(3): 687-697.

[10] 周新宇，吴志健，王明文. 基于正交实验设计的人工蜂群算法 [J]. 软件学报, 2015, 26(9): 2167-2190. (Zhou Xinyu, Wu Zhijian, Wang Mingwen. Artificial bee colony algorithm based on orthogonal experimental design [J]. Journal of Software, 2015, 26(9): 2167-2190.)

[11] Civicioglu P. Backtracking search optimization algorithm for numerical optimization problems [J]. Applied Mathe-Matics& Computation, 2013, 219(15): 8121-8144.

[12] 田文凯，刘三阳，王晓娟. 基于差分进化的回溯搜索优化算法研究与改进 [J]. 计算机应用研究, 2015, 32(6): 1653-1656. (Tian Wenkai, Liu Sanyang, Wang Xiaojuan. Study and improvement of backtracking search optimization algorithm based on differential evolution [J]. Application Research of Computers, 2015, 32(6): 1653-1656.)

[13] 李牧东，赵辉，翁兴伟. 具有广泛学习策略的回溯搜索优化算法 [J]. 系统工程与电子技术, 2015, 37(4): 958-963. (Li Mudong, Zhao Hui, Weng Xingwei. Backtracking search optimization algorithm with comprehensive learning strategy [J]. Systems Engineering and Electronics, 2015, 37(4): 958-963.)

[14] Chen Debao, Zou Feng, Lu Renquan, *et al*. Learning backtracking search optimisation algorithm and its application [J]. Information Sciences, 2017, 376: 71-94.

[15] Mariot L, Picek S, Jakobovic D, *et al*. Evolutionary algorithms for the design of orthogonal latin squares based on cellular automata [C]// Proc of Genetic and Evolutionary Computation Conference.New York: ACM Press, 2017: 306-313.

[16] 贺智明，宋建国，梅宏标. 结合元胞自动机的果蝇优化算法 [J]. 计算机应用, 2014, 34(8): 2295-2298. (He Zhiming, Song Jianguo, Mei Hongbiao. Fruit fly optimization algorithm based on cellular automata [J]. Journal of Computer Applications, 2014, 34(8): 2295-2298.)

[17] 程健，金菊良，周玉良，等. 基于正交试验和元胞自动机模型的加速并行遗传算法[J].系统工程理论方法应用, 2006(4): 364-367. (Cheng Jian, Jin Juliang, Zhou Yuliang, *et al*. Orthogonal design and cellular automata based acceleration parallel genetic algorithm [J]. Systems Engineering-Theory Methodology Applications. 2006(4): 364-367.)

[18] 蔡自兴，江中央，王勇，等. 一种新的基于正交实验设计的约束优化进化算法 [J]. 计算机学报, 2010, 33(5): 855-864. (Cai Zixing, Jiang Zhongyang, Wang Yong, *et al*. A novel constrained optimization evolutionary algorithm based on orthogonal experimental design [J]. Chinese Journal of Computers, 2010, 33(5): 855-864.)

[19] 李康顺，左磊，李伟. 基于单形正交实验设计的差分演化算法 [J]. 计算机应用, 2016, 36(1): 143-149. (Li Kangshun, Zuo Lei, Li Wei. Novel differential evolution algorithm based on simplex-orthogonal experimental design [J]. Journal of Computer Applications, 2016, 36(1): 143-149.)

[20] Abunaser A, Alshattnawi S. Hybridizing artificial bee colony algorithm with multi-parent crossover operator [J]. International Journal of Applied Metaheuristic Computing, 2015, 6 (2): 608-614.

[21] 张锦华，宋来锁，张元华，等. 加权变异策略动态差分进化算法 [J]. 计算机工程与应用, 2017, 53(4): 156-162. (Zhang Jinhua, Song Laisuo, Zhang Yuanhua, *et al*. Dynamic differential evolution algorithm with weighted mutation strategy [J]. Computer Engineering and Applications, 2017, 53 (4): 156-162.)